Semiannual Status Report for NAG 5–1612

Submitted to:
Instrument Division
Engineering Directorate
Goddard Space Flight Center
Greenbelt, MD 20771
Attn: Warner Miller and Pen Shu Yeh

K. Sayood, Y.C. Chen and X. Wang

Department of Electrical Engineering
and
Center for Communication & Information Science
University of Nebraska-Lincoln
Lincoln, Nebraska 68588–0511

Period: December 15, 1991 — June 15, 1992

1 Introduction

During this reporting period we have worked on three somewhat different problems. These are modeling of video traffic in packet networks, low rate video compression and the development of a lossy + lossless image compression algorithm, which might have some application in browsing algorithms. The lossy + lossless scheme is an extension of work previously done under this grant. it provides a simple technique for incorporating browsing capability. The low rate coding scheme is also a simple variation on the standard DCT coding approach. In spite of its simplicity the approach provides surprisingly high quality reconstructions. The modeling approach is borrowed from the speech recognition literature, and seems to be promising in that it provides a simple way of obtaining an idea about the second order behavior of a particular coding scheme. Details about these are presented in the following sections.

2 Lossy+Lossless Compression

Lossless compression of images consist of two steps; a decorrelation step in which the redundancies within the image are exploited to reduce the first order entropy of the image, and a coding step in which variable length codes are used to provide coding rates close to the entropy. The second step has been very well studied with the development of coding scheme for sources with known statistics, such as the Huffman codes [1] and Arithmetic codes. More recently *universal coding* schemes such as the Rice algorithm [2] have been developed for coding sources with unknown statistics. The problem of decorrelation is not that well studied, and to date the best decorrelation strategies have been predictive techniques.

The recently proposed JPEG still compression standard [3] uses predictive techniques to decorrelate the image. It provides eight different predictive schemes from which the user can select. Table 1 lists the eight predictors. The first scheme makes no prediction. The next three are one-dimensional predictors and the last four are two-dimensional prediction schemes.

Sayood and Anderson [4] propose a *switched prediction scheme* which has static ordering and replacement functions and a backward adaptive neighborhood function. They scan the image in raster order, predicting the value

Mode	Prediction for $P[i,j]$
0	0 (No Prediction)
1	$P[i-1,j]$
2	$P[i,j-1]$
3	$P[i-1,j-1]$
4	$P[i,j-1] + P[i,j-1] - P[i-1,j-1]$
5	$P[i,j-1] + (P[i,j-1] - P[i-1,j-1])/2$
6	$P[i-1,j] + (P[i-1,j] - P[i-1,j-1])/2$
7	$(P[i,j-1] + P[i-1,j])/2$

Table 1: JPEG Predictors for lossless coding

of the current pixel by using a reference pixel (say, the left neighbor). If the prediction error exceeds a certain threshold then the reference pixel for the next pixel is switched (say, to the top neighbor). The scheme is very simple and can be implemented efficiently in hardware.

Another simple but perhaps more effective technique, named MAP (Median Adaptive Prediction), is given by Martucci [5]. Here, the median of a set of predictions is chosen as the prediction that is used to form the prediction error. Simulations were reported using the median of the following three predictors

1. $P[i,j-1]$
2. $P[i-1,j]$
3. $P[i,j-1] + P[i-1,j-1]$ - $P[i-1,j-1]$

Results obtained were an improvement over any of the three predictors taken individually. The reason for this is that the median adaptive predictor would always choose either the best or the second best predictor among the candidate predictors.

Given the success of lossy image compression techniques at generating an excellent visual approximation of an image at very low bit rates, the following scheme seems a natural candidate:

- First generate a low bit rate representation of the image by some lossy technique.

- Use this low bit rate approximation to decorrelate the image by forming a residual which represents the difference between the original image and the low rate approximation.

Such schemes are called *Lossy plus Lossless schemes.*

In order to reconstruct the image from the residual, the receiver would need to first have the low rate approximation. So one can see that in effect we have a decorrelation technique with a forward adaptive replacement function. Techniques that use a discrete cosine transform based lossy step have been investigated in [6]. Using the Walsh-Hadamard transform and S-transform in the lossy step was investigated in [7]. Also, investigated in the same study was Subband Coding using the Smith and Barnwell Filter as well as the Quadrature Mirror Filter for getting a low bit rate approximation of the image. Manohar and Tilton [8] give a Vector Quantization based lossy plus lossless technique. They get improved performance by iterating this process again on the residual by using a special codebook for the residual image. They report best performance for three such iterations.

One advantage of lossy plus lossless techniques is that they provide the user with a low rate approximation of an image, based on which the decision for viewing the exact image can be made. This is generally called *browsing capability*. It finds applications in situations where the user may have to scan through a large database of images in order to find a specific image of interest. The disadvantage is that the final bit-rate is generally higher than the bit rate that would have been obtained if the image had been losslessly coded directly, instead of first going through the lossy encoding step [9].

Although a variety of schemes exist for image decorrelation, very few comparative studies have been reported in literature. A performance comparison of some of the schemes listed above is given in [7] for medical images. It was concluded in this study that the HINT scheme was more effective then the other schemes studied. However, in [9], it was observed that predictive techniques out perform other techniques.

In this work, we shall take the JPEG still image compression standard [3] as a basis for comparison. We do so because our experience has shown the scheme to be quite robust and yields superior performance over a wide range of images. We have chosen a set of test images (given in appendix 1) on which the eight different predictors listed by JPEG were tried. Table 2 lists the entropy of the residual image for the test images.

Image	JPEG 0	JPEG 1	JPEG 2	JPEG 3	JPEG 4	JPEG 5	JPEG 6	JPEG 7
USC-Girl	6.42	5.05	5.10	5.40	5.07	4.90	4.93	4.82
Girl	6.49	4.25	4.37	4.65	4.68	4.68	4.75	4.53
Lady	5.37	3.83	4.16	4.31	4.09	3.81	4.02	3.84
House	6.54	4.64	5.06	5.35	4.58	4.46	4.64	4.64
Couple	5.96	4.67	4.49	5.11	4.36	4.38	4.27	4.41
Tree	7.21	5.63	5.93	6.04	5.74	5.49	5.66	5.51
Satellite	7.31	6.15	6.39	6.55	6.09	5.90	6.01	5.89

Table 2: Entropy of error image using JPEG predictors

In our technique we used a lossy compression scheme developed under a grant from the Goddard Space Flight Center (NAG 5-916). The details of the lossy scheme is described in a recently published paper [10], a copy of which is included. The heart of this scheme is a recursively indexed quantizer [4] which maps a large (possibly countably infinite) set into a small finite set. This means that the entropy coding that is to be performed can be done on a small alphabet, which can result in substantial savings in hardware complexity. In our implementation the size of the output alphabet varied from three to nine. This can be contrasted with the JPEG lossless scheme where the size of the input alphabet of the entropy coder (output alphabet of the decorrelation scheme) is 511 (this could be reduced to 256 by being somewhat clever about how to store the residuals).

The scheme works as follows: the Edge Preserving DPCM (EPDPCM) scheme is first used to encode the image at some required fidelity level. If a lossless version is then required the difference between the reconstructed image and the original is then transmitted to the receiver. One of the attractive features of the EPDPCM scheme is that the reconstruction error can be strictly limited to within a predetermined limit. Thus, we could encode the image so that the error is confined to the least significant bit, or the least m significant bits. This makes the lossless step very simple. Depending on the fidelity of the lossy step, we could use m bits, without the need for any further entropy coding.

We tried a variety of predictors in the EPDPCM scheme. The two that

	$\Delta = 2$			$\Delta = 4$			$\Delta = 8$		
Image	Lossy	Lossless	Total	Lossy	Lossless	Total	Lossy	Lossless	Total
USC-Girl	3.93	1	4.93	2.84	2	4.84	1.95	3	4.95
Girl	3.91	1	4.91	2.84	2	4.84	1.97	3	4.97
Lady	3.14	1	4.14	2.17	2	4.17	1.52	3	4.52
House	3.75	1	4.75	2.74	2	4.74	1.92	3	4.92
USC-Couple	3.61	1	4.61	2.57	2	4.57	1.73	3	4.73
Tree	5.22	1	6.22	3.74	2	5.74	2.79	3	5.79
Satellite	5.49	1	6.49	3.95	2	5.95	2.94	3	5.94

Table 3: Rates for the Lossy + Lossless Scheme Using the Harrison Predictor

gave the best results were a predictor due to Harrison [11], and variation of the MAP predictor [5]. The Harrison predictor is of the form $\frac{2}{3}P(i,j-1) + \frac{2}{3}P(i-1,j) - \frac{1}{3}P(i-1,j-1)$. The results are shown in Table 3.

In these simulations we used a nine level recursively indexed quantizer. The best results seem to occur for a step size (Δ) of four. The final lossless performance is within .3 bits of the best JPEG lossless scheme in each case. For the USC-Girl and Satellite images, the lossy+lossless scheme actually performs as well as the JPEG schemes.

We also simulated the median adaptive predictor with one slight modification. In the published form the MAP has infinite memory. This makes it unsuitable for use in lossy schemes, as the quantization error will tend to propagate. We therefore multiplied the prediction with a prediction coefficient of 0.85. This makes the predictor leaky and allows the effect of the errors to die out over time. The results are presented in Table 4

As the MAP predictions are somewhat better than the predictions from the Harrison predictor we used a three level recursively indexed quantizer for all but the Tree image. The best results in these simulations seem to be obtained when Δ has a value of two(except for the Satellite image). Notice that in this case the performance for some of the images is actually *better* than the performance of the JPEG lossless scheme.

We have presented a simple Lossy + Lossless compression scheme which compares favorably with existing schemes in terms of the bit rate. However,

	$\Delta = 2$			$\Delta = 4$			$\Delta = 8$		
Image	Lossy	Lossless	Total	Lossy	Lossless	Total	Lossy	Lossless	Total
USC-Girl	3.51	1	4.51	2.60	2	4.60	1.93	3	4.93
Girl	3.41	1	4.41	2.70	2	4.70	2.12	3	5.12
Lady	3.38	1	4.38	2.69	2	4.69	2.06	3	5.06
House	3.60	1	4.60	2.81	2	4.81	2.14	3	5.14
USC-Couple	3.40	1	4.40	2.49	2	4.49	1.79	3	4.79
Tree	5.67	1	6.67	4.47	2	6.47	3.36	3	6.36
Satellite	5.26	1	6.26	3.82	2	5.82	2.74	3	5.74

Table 4: Rates for the Lossy + Lossless Scheme Using the Median Adaptive Predictor

to be truly competitive, the first lossy pass should have a significantly lower bit rate, to accomodate quick previews. This could be done by subsampling the image first and providing a coded version of the subsampled image to the user. We are currently working on this problem.

3 Low Rate Video Coding

Xiaomei Wang

Transform coding is a widely accepted method for image and video compression. The basic motivation behind transform coding is to remove the source redundancy by decomposing the input signal into components in the frequency or transform domain, i.e. translate a set of data into another set of less correlated or more independent coefficients. Of particular interest to image processing and image coding standards is the two dimensional discrete cosine transform. The DCT provides a good match to the optimum (covariance-diagonalizing or Karhunen-Loeve) transform for most image signals and fast algorithm exist for computing the DCT.

Traditionally as well as for convenience it is assumed that all of the important coefficients are packed into a specific area of the transform domain, this is called the "energy compaction" effect of the cosine transform. The amount of compression depends upon the number of coefficients retained in

-4	11	-5	-8	-8	-1	9	-7
4	2	-5	-5	1	-4	-8	2
-5	-7	5	-3	0	4	-2	2
-8	-7	3	-3	4	6	-8	-14
1	-3	0	-3	3	2	-2	-5
-6	-4	-6	5	-6	-5	-9	7
8	-2	-1	-6	3	5	6	2
0	-1	-6	-3	0	-8	-8	3

Table 5: Block of difference image

this area. Usually the low frequency area is considered more important than the high frequency area. For this reason image data is often compressed by coding and then transmitting only the low-frequency components. But this assumption is not always true.

Another possibility is to put a threshold on the transformed coefficient magnitude and set all coefficients with magnitudes below the threshold to zero. This approach is more realistic because we do not assume any fixed important area but consider this area dynamic, depending on the characteristics of images. This is a more complex coding strategy but it results in a very high compression rate while maintaining better picture quality, i.e. more details and less block effect compared to coding and transmitting only the low frequency components.

In the following we describe a threshold transform coding scheme which incorporates DPCM and runlength coding. The motion picture sequence used for testing is that of a woman talking on the phone. This is one of the standard sequences used by the MPEG committee.

In Figure 1 we show one of the frames from this sequence. The difference image between the current frame and the quantized version of last frame is shown in Figure 2. We will divide this image into N by N sub-blocks and process each block separately. Let us take a look at a randomly chosen 8x8 block shown in Table 5.

After the cosine transform the coefficients are shown in Table 6

We can see that the there is no obvious energy compaction for this block and comparatively larger values are scattered around the block. We can see

-3.0	-0.1	0 0	1.0	0.0	-1.4	0.8	0.2
-0.3	0.5	-0.2	0.3	-2.9	-0.2	-0.8	0.0
0.4	0.4	4.0	-0.1	-0.1	-0.5	-2.7	-0.3
0.3	-0.0	1.7	-0.1	-0.8	0.6	-2.1	-1.4
-1.3	1.2	-1.3	1.0	-2.1	-0.0	-1.3	0.4
0.8	-0.9	-1.0	1.3	-3.1	1.2	0.1	0.6
-1.3	-1.2	-0.2	-2.9	0.0	0.2	-0.2	-1.5
2.3	-0.7	1.2	1.1	-1.4	0.9	0.1	1.4

Table 6: DCT coefficients

-3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	4.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	-3.1	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Table 7: Coefficients with threshold = 3

this more clearly by using the threshold strategy. We choose a threshold t as a positive number and compare each coefficient in the transform domain with the threshold. If the magnitude of the coefficient is less than the threshold, we set it to zero; if it is larger than or equal to the threshold, we retain it without change.

With t = 3, we get the block in Table 7.

We can see there are only three non-zero coefficients left, not all of them are in the low frequency area. However, in spite of the fact that we have only three of the original sixty four coefficients left we will see that after the inverse transform we can still get a very good looking picture. The image in Figure 3 was reconstructed after zeroing out all coefficients with a magnitude less than three. As can be seen this is a very good reproduction of the original image with only $\frac{3}{64}$ of the original data. The reconstructed image with threshold = 5 is shown in Figure 4.

The setting of the threshold depends upon the compression as well as the picture quality we need. The amount of compression can be somehow indicated by the number of non-zero coefficients left. The number of non-zero coefficients as a function of the threshold is shown in Figure 5. As to the picture quality, after doing the inverse cosine transform and DPCM decoding we compare the image with the original and compute the PSNR. The PSNR as a function of the threshold is shown in Figure 6.

To transfer the block of coefficients we first linearize the two dimensional block along a zig-zag scanning path as shown in Figure 7 and get the sequence of data containing almost all zeros except for a few non-zero data. We will use run length coding and Huffman code for such data.

The non-zero coefficients are quantized. The quantizer is designed based on the probability distribution of data. We show the relative distribution of frame 25 in Figure 8. Here we choose threshold t = 5, so the center of x-coordinate is +5 or -5. The distribution of data from the other frames is similar.

We design our quantizer based on this distribution however as there are variations from frame to frame, we choose to have more output levels in case these are needed in some other frames. Since we use Huffman code for the quantizer outputs, the bit rate will not increase much because of more quantizer outputs. The Huffman code we use for non- zero coefficients is in Table 8.

The relative distribution of runs and the Huffman code designed for the runs is shown in Table 9

In order to reduce the number of bits we will not count the last zero run of each block. In order to do this we need a symbol for either the beginning or the end of the block. We will count the number of runs of each block and send it as the header of each block. The relative probability distribution of number of runs of each block is shown in table 10, again we use an entropy code for coding efficiency.

Because the distribution of each frame varies, there is no need to design Huffman code exactly according to one frame. The code in the table is comparatively easy and efficient.

Using the above methods the final bit rate for frame 25 is about 0.24

Number of Outputs	Code	
0	63340	No Code
1	1272	1
2	763	01
3	115	001
4	42	0001
5	4	00001
6	0	000001
7	0	0000001
8	0	00000001

Table 8: Quantizer outputs statistics and Huffman codes

Run Length	Number of Runs	Code
0	233.000000	1110 (¿15)
1	1651.000000	0
2	477.000000	110
3	184.000000	1001
4	151.000000	1000
5	120.000000	10111
6	94.000000	10101
7	77.000000	111111
8	73.000000	111110
9	62.000000	111101
10	50.000000	101001
11	61.000000	111100
12	52.000000	101100
13	44.000000	101000
14	35.000000	1011011
15	18.000000	1011010

Table 9: Relative distribution of runs and Huffman codes

Number of Coefficients	Occurrence	Code
0	445	1
1	163	01
2	106	001
4	83	0001
5	70	00001
5	57	000001
6	37	0000001
7	30	00000001
8	11	000000001
9	12	0000000001
10	3	00000000001
11	4	000000000001
12	2	0000000000001
13	0	00000000000001
14	1	000000000000001

Table 10: Distribution of number of coefficients in each block and a trivial entropy code

(bits/pixel). The reconstructed image after coding is shown in Figure 9. Using the same code for other frames the bit rate varies a little but not too much.

We have described a simple easy to implement low rate video coding scheme. To keep the algorithm simple we have not used any motion compensation or more complicated quantization techniques. The thresholding operation can be made simpler if we chose the threshold to be a power of two. In this case the thresholding would simply consist of shifting the least significant bits out.

4 Using Hidden Markov Model as Video Source Output Model

Yun-Chung Chen

4.1 Introduction

Variable bit rate coding scheme will be implemented in ATM networks in order to obtain flexibility and efficiency. This is important because the output bit rate stream of a video source depends on the specific scene contents and coding scheme used. Also, different types of video sources will have different statistical characteristics, and different bit rates. Thus it would be inefficient to use fixed rate coding schemes. In this project we examine the CCITT H.261 coding scheme which is a proposed standard for video-telephony or single-activity motion scenes. We are interested in how to transmit the coded video information across ATM networks efficiently. Performance simulations are very important when designing a coding scheme which will hopefully best fit into the future ATM environment. Efficient and accurate simulations depend on accurate modeling. Unfortunately, the modeling of video sources is more complicated than the voice source model like Modulated Markov Poisson Process(MMPP) [12]. The use of continuous state autoregressive processes used for video source modeling usually generates significant difficulties in any analytical analysis. Maglaris et. al. [13] develop a discrete state, continuous time Markov process to simplify the analysis. In this project, we use the concept of Hidden Markov Models(HMM) to simulate the variable output rate of a video source. Whenever the analytical analysis is impossible, we hope to get some insight about the performance from the simulation. Even when the analytical analysis is possible, we can have a comparison tool. We show that the HMM as a video source output model can accurately reflect atleast the second order video output statistics.

4.2 Problem Setup

An HMM is characterized by the following:

1. N, the number of states in the model.
2. M, the number of distinct observation symbols per state.
3. $\mathbf{A} = \{a_{ij}\}$, the state transition probability distribution.
4. $\mathbf{B} = \{b_j(k)\}$, the observation symbol probability distribution.
5. $\pi = \{\pi_i\}$, the initial state distribution.

Most video source models developed have used the frame as a unit when modeling the output sequence [13]. Considering the data structure used in the H.261 coding algorithm, we decided to use a macroblock (16 x 16 pixels) as our unit simply because the coding algorithm adopts different quantization strategies for every macroblock. H.261 changes the step size of quantizer depending on the buffer fullness. If the buffer is full, coarse quantization will produce less output and release the tight condition of buffer. Using the quantization mode as the state in the HMM seems to be a natural choice. We hope this choice can accurately reflect the bit rate distribution in different quantization modes. As a test sequence we used the *Susie* sequence, which is one of the standards from MPEG. We developed an H.261 simulator and used frames 46-55 of the *Susie* sequence. The choice of these 10 frames is based on the consideration of covering the states, symbols and state transitions adequately. Using these ten frames gave us 2560 output bit rates. Through the coding simulation of these ten frames, the quantization step travels back and forth in the set:(8,16,24,32,40,48,56). Each quantization step denotes a state in our model, therefore N equals 7. The coder generates a zero output when dealing with a motionless macroblock, and when the bufffer is full. So, 0 is assigned as a distinct symbol. The observation symbols are the output of the quantizer. We use an eight level quantizer so M equals eight.

0.800000	0.200000	0.000000	0.000000	0.000000	0.000000	0.000000
0.100000	0.800000	0.100000	0.000000	0.000000	0.000000	0.000000
0.000000	0.100000	0.800000	0.100000	0.000000	0.000000	0.000000
0.000000	0.000000	0.100000	0.800000	0.100000	0.000000	0.000000
0.000000	0.000000	0.000000	0.100000	0.800000	0.100000	0.000000
0.000000	0.000000	0.000000	0.000000	0.100000	0.800000	0.100000
0.000000	0.000000	0.000000	0.000000	0.000000	0.200000	0.800000

Table 11: Initial condition A.opt for the transition probability matrix

0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857
0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857

Table 12: Initial condition A.uni for the transition probability matrix

There are several possible ways of initializing the algorithm used for developing the Hidden Markov Model. These depend on the selection of the initial state transition matrix **A**, the matrix of observation symbol probability distributions **B**, and the initial state probability distribution π. The different initial values of these matrices used in this work are shown in Table 11 to Table 16. Eight different combinations of these initial parameters were used to run the optimization. *.uni is uniform distribution for **A**, **B** and π matrix. A.opt is the approximate form we think the **A** matrix should take since one can only travel between neighboring states. B.opt is actually calculated from the H.261 simulation, therefore we can comfortably assume it is optimal. The initial state probability matrix π.opt is obviously correct since we start the simulation with an empty buffer.

0.160000	0.240000	0.080000	0.120000	0.080000	0.040000	0.040000	0.240000
0.155555	0.133333	0.266666	0.222222	0.133333	0.044444	0.022222	0.022222
0.187214	0.109589	0.420091	0.178082	0.082191	0.013698	0.004556	0.004556
0.277456	0.057803	0.421905	0.173410	0.052023	0.005780	0.005780	0.005780
0.223034	0.113528	0.451553	0.138939	0.027422	0.020109	0.003656	0.000000
0.225769	0.125427	0.461104	0.114025	0.041049	0.022805	0.004561	0.000000
0.186943	0.154302	0.528189	0.062314	0.050445	0.016320	0.001483	0.000000

Table 13: Initial condition B.opt for the observation probability matrix

0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000
0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000	0.125000

Table 14: Initial condition B.uni for the observation probability matrix

1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000

Table 15: Initial condition π.opt for the initial probability matrix

0.142857	0.142857	0.142857	0.142857	0.142857	0.142857	0.142857

Table 16: Initial condition π.opt for the initial probability matrix

0.971572	0.028428	0.000000	0.000000	0.000000	0.000000	0.000000
0.000000	0.723474	0.221840	0.006125	0.047594	0.000935	0.000044
0.000000	0.074619	0.880526	0.044584	0.000242	0.000000	0.000000
0.000000	0.004280	0.108101	0.869359	0.000116	0.001380	0.016837
0.000000	0.000000	0.000000	0.290687	0.705150	0.003746	0.000020
0.000000	0.000000	0.000000	0.000000	0.080689	0.858148	0.061029
0.000000	0.000000	0.000000	0.000000	0.000000	0.095528	0.904163

Table 17: **A** matrix at 29th iteration using A.opt B.opt and π.opt

0.272096	0.255770	0.106084	0.113691	0.056853	0.000000	0.024947	0.170560
0.009124	0.000000	0.085811	0.399809	0.325566	0.129653	0.040775	0.009262
0.008507	0.129088	0.760925	0.101409	0.000071	0.000000	0.000000	0.000000
0.897719	0.101437	0.000844	0.000000	0.000000	0.000000	0.000000	0.000000
0.014575	0.638788	0.346638	0.000000	0.000000	0.000000	0.000000	0.000000
0.000000	0.026007	0.436730	0.361496	0.174790	0.000000	0.000977	0.000000
0.021606	0.154724	0.814827	0.000000	0.000000	0.000000	0.008843	0.000000

Table 18: **B** matrix at 29th iteration using A.opt B.opt and π.opt

4.3 Discussion

Contrary to our expectations, using A.opt, B.opt and π.opt as initial parameters didn't generate the optimal solution. Instead, the combination of A.uni, B.opt and π.uni produced the highest score $P(o|\lambda)$. This is not that surprising if we consider that A.opt actually is not optimal. Some sample values of **A**, **B**, and π are shown in Tables 17 through 22.

Because of the length of the sequence (2560) used in the optimization procedure, we ran into the problem of underflow. We used a scaling algorithm to correct most of the effects of the underflow. However, this was not sufficient, and some elements in the **B** and π matrix sometimes got very small and went to zero during the simulation. This caused problems, as computation of the path metric requires the computation of logarithms of the probabilities. Therefore, whenever some number is got very small, we artificially set

1.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000

Table 19: π matrix at 29th iteration using A.opt B.opt and π.opt

0.596075	0.231178	0.002032	0.000074	0.008604	0.029235	0.132831
0.112090	0.531015	0.034520	0.013038	0.003539	0.003081	0.302962
0.005984	0.294194	0.518999	0.072965	0.000175	0.000119	0.108050
0.004642	0.012775	0.079244	0.823574	0.023301	0.056420	0.000001
0.000661	0.063367	0.003103	0.446239	0.405644	0.003523	0.077228
0.000006	0.021556	0.038011	0.104896	0.138746	0.150711	0.546299
0.013463	0.067384	0.220619	0.000094	0.026825	0.143878	0.527727

Table 20: **A** matrix at 136th iteration using A.uni B.opt and π.opt

0.000000	0.000000	0.000000	0.140243	0.506068	0.229788	0.072147	0.051754
0.012791	0.013442	0.317190	0.565088	0.091490	0.000000	0.000000	0.000000
0.000000	0.050092	0.949505	0.000402	0.000000	0.000000	0.000000	0.000000
0.976901	0.023099	0.000001	0.000000	0.000000	0.000000	0.000000	0.000000
0.280123	0.719769	0.000108	0.000000	0.000000	0.000000	0.000000	0.000000
0.061535	0.928222	0.010243	0.000000	0.000000	0.000000	0.000000	0.000000
0.003591	0.092106	0.895200	0.000791	0.000000	0.003086	0.005225	0.000000

Table 21: **B** matrix at 136th iteration using A.uni, B.opt, and π.uni

1.000000
0.000000
0.000000
0.000000
0.000000
0.000000
0.000000

Table 22: π matrix at 136th iteration using A.uni, B.opt, and π.uni

it to a constant(10^{-100}). This probably would not affect the optimal path tracking because these small numbers are in the range of 10^{-100} and the paths using these small number would not be selected anyway. But it does affect the reestimation procedure slightly, in that the probability distribution won't sum up exactly to 1.

The optimal **B** matrix we get is not very close to bmax.opt which is actually observed from the experiment. But it still reflects the fact that coarse quantization is more likely to produce small output. The average value μ over all 10 frames and the standard deviation σ were found to be $\mu = 26.05$ bits/macroblock and $\sigma = 22.98$ bits/macroblock. Using HMM, we generate a output sequence with $\mu = 26.44$ bits/macroblock and $\sigma = 21.99$ bits/macroblock, which is pleasantly close to our original data. Furthermore, we calculated the autocorrelation

$$\frac{C(\tau)}{C(0)} = \frac{E[\lambda(t)\lambda(t+\tau)] - \mu^2}{C(0)} \tau = 1, 2, ..., 256$$

Although the values are not close for both sequences, they appear to have almost the same shape (Figures 10 and 11). It is nice to notice the model generates similar correlation structure as the H.261 output since variance and covariance values usually dominate the queuing behavior. It should be noted that we are mostly interested in the correlation behavior for small lags, as these values are used in the queuing analysis. The 10 frame sequence from *Susie* generates a lot of motion, as the woman is shaking her head. For those frames without this much motion, quantization step won't go that high. It means the hidden Markov Model developed here for the high-motion sequence probably is not suited for a still sequence. Adopting the idea from MMPP, we can develop another hidden Markov model with different mean and variance for motionless sequence. And then build another Markov chain to change the models (high/low motion) alternatively in the simulation.

References

[1] D.A. Huffman. A Method for the Construction of Minimum Redundancy Codes. *Proceedings of the IRE*, 40:1098–1101, September 1952.

[2] R.F. Rice. Some practical universal noiseless coding techniques. Technical Report JPL Publication 79-22, NASA, 1979.

[3] G. K. Wallace. The JPEG still picture compression standard. *Communications of the ACM*, 34(4):31–44, April 1991.

[4] K. Sayood and K. Anderson. A differential lossless image compression algorithm. *IEEE Transactions on Signal Processing*, 40(1):236–241, January 1992.

[5] S. A. Martucci. Reversible compression of HDTV images using median adaptive prediction and arithmetic coding. In *IEEE International Symposium on Circuits and Systems*, pages 1310–1313. IEEE Press, 1990.

[6] J. R. Cox, S. M. Moore, G. J. Blaine, J. B. Zimmerman, and G. K. Wallace. Optimization of trade-offs in error-free image transmission. In *Proceedings SPIE Medical Imaging III, 1091*, pages 1285–1292. SPIE, 1989.

[7] P. Roos, A. Viergever, M. C. A. Van Dijke, and J. H. Peters. Reversible intraframe compression of medical images. *IEEE Transactions on Medical Imaging*, 7(4):328–336, December 1988.

[8] M. Manohar and J. C. Tilton. Progressive vector quantization of multispectral image data using a massivley parallel SIMD machine. In J. A. Storer and M. C. Cohn, editors, *Proceedings of the Data Compression Conference*, pages 181–190. IEEE Computer Society Press, 1992.

[9] M. Rabbani and P.W Jones. *Digital Image Compression Techniques*, volume TT7 of *Tutorial Texts Series*. SPIE Optical Engineering Press, 1991.

[10] M.C. Rost and K. Sayood. An Edge Preserving Differential Image Coding Scheme. *IEEE Transactions on Image Processing*, 1:250–256, April 1992.

[11] C. W. Harrison. Experiments with linear prediction in television. *Bell System Tech. J.*,, 31:764–783, July 1952.

[12] H. Heffes and M. Lucantoni. A Markov Modulated Charecterization of Packetized Voice and Data Traffic and Related Multiplexer Performance. *IEEE Journal on Selected Areas of Communications*, September 1986.

[13] W.B. Maglaris et. al. Performance Models of Statistical Multiplexing in Packet Video Communications. *IEEE Transactions on Communications*, July 1988.

ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Figure 1.
Original image from the *Susie* sequence (frame 25)

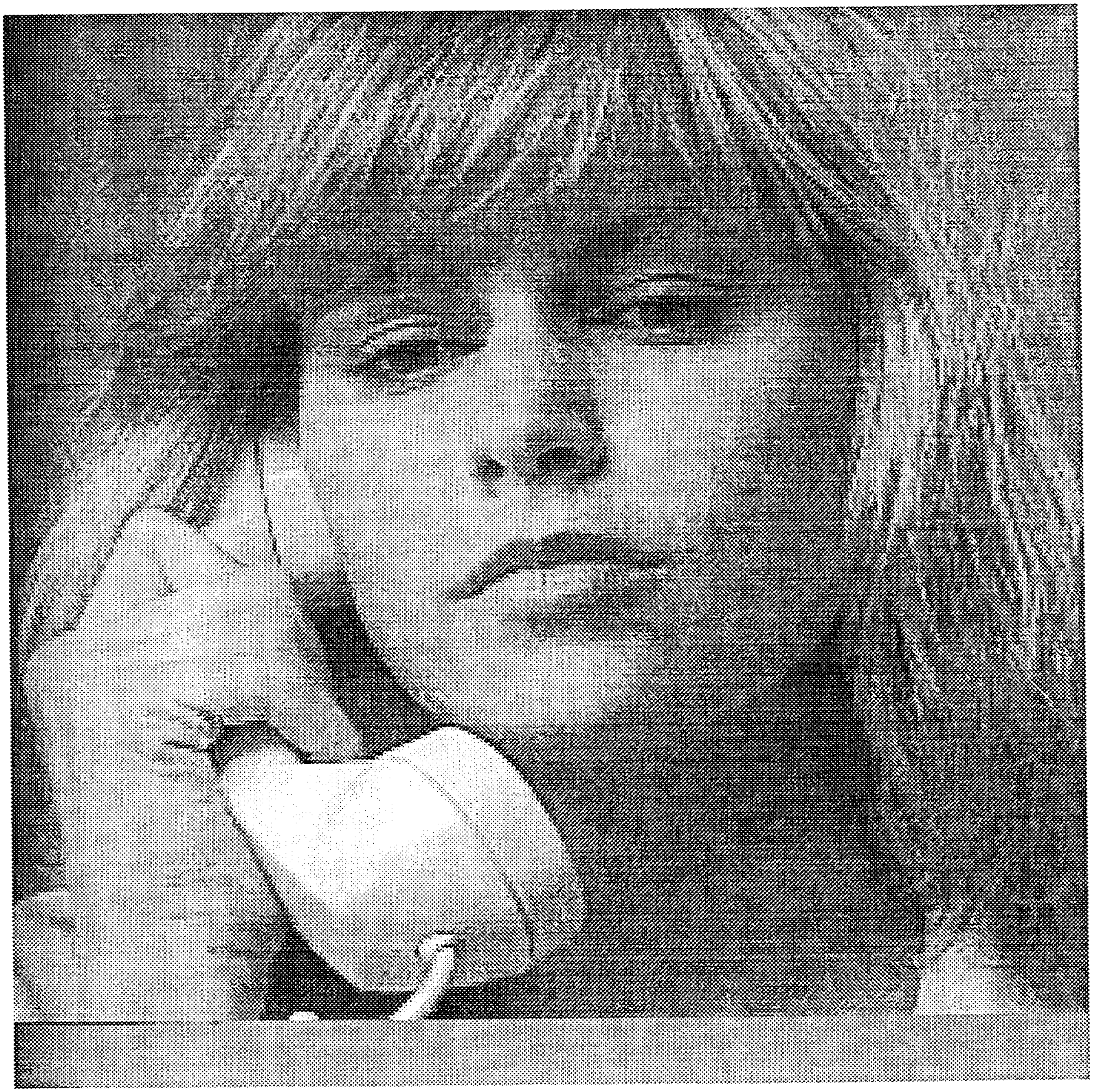
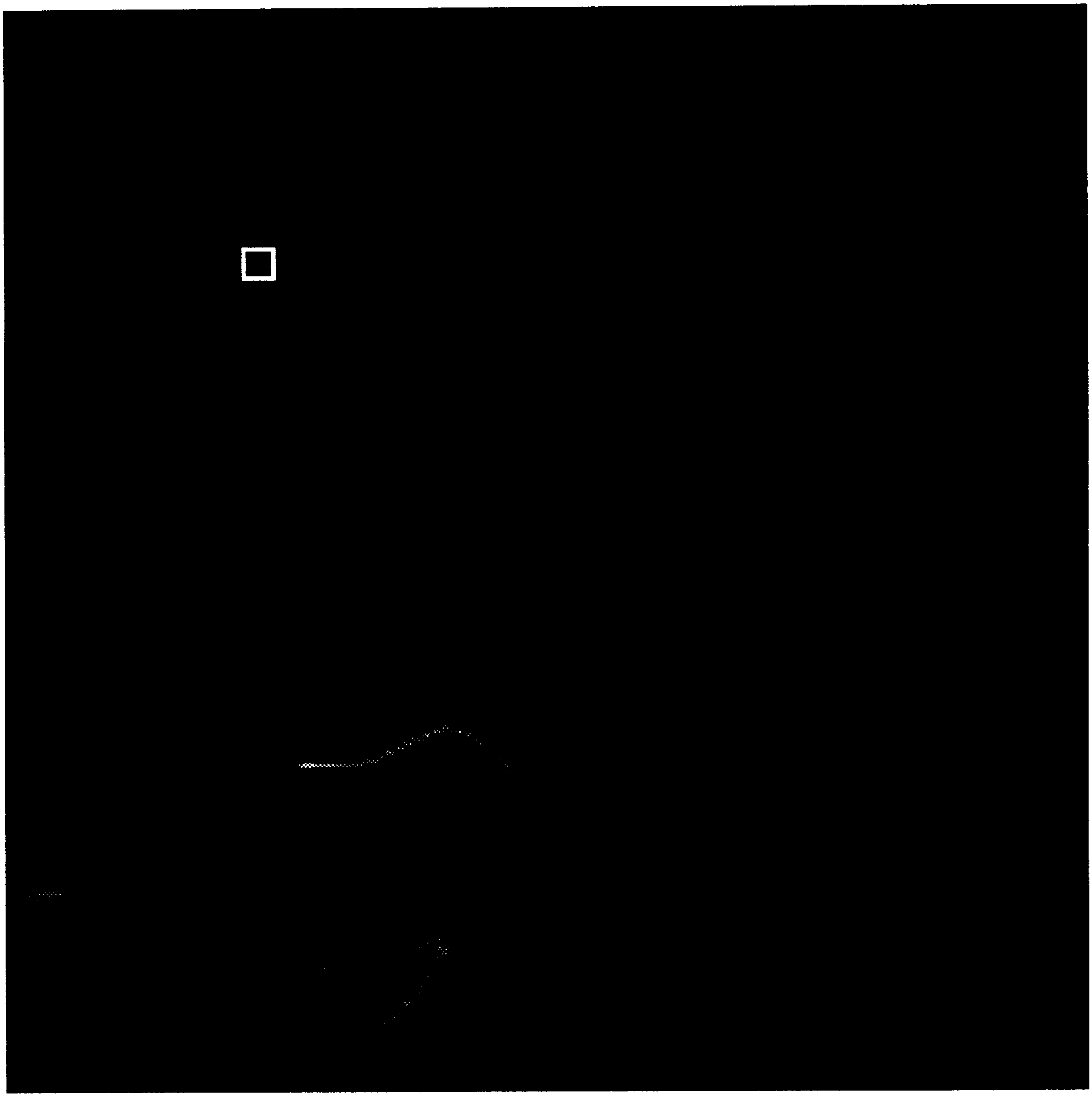
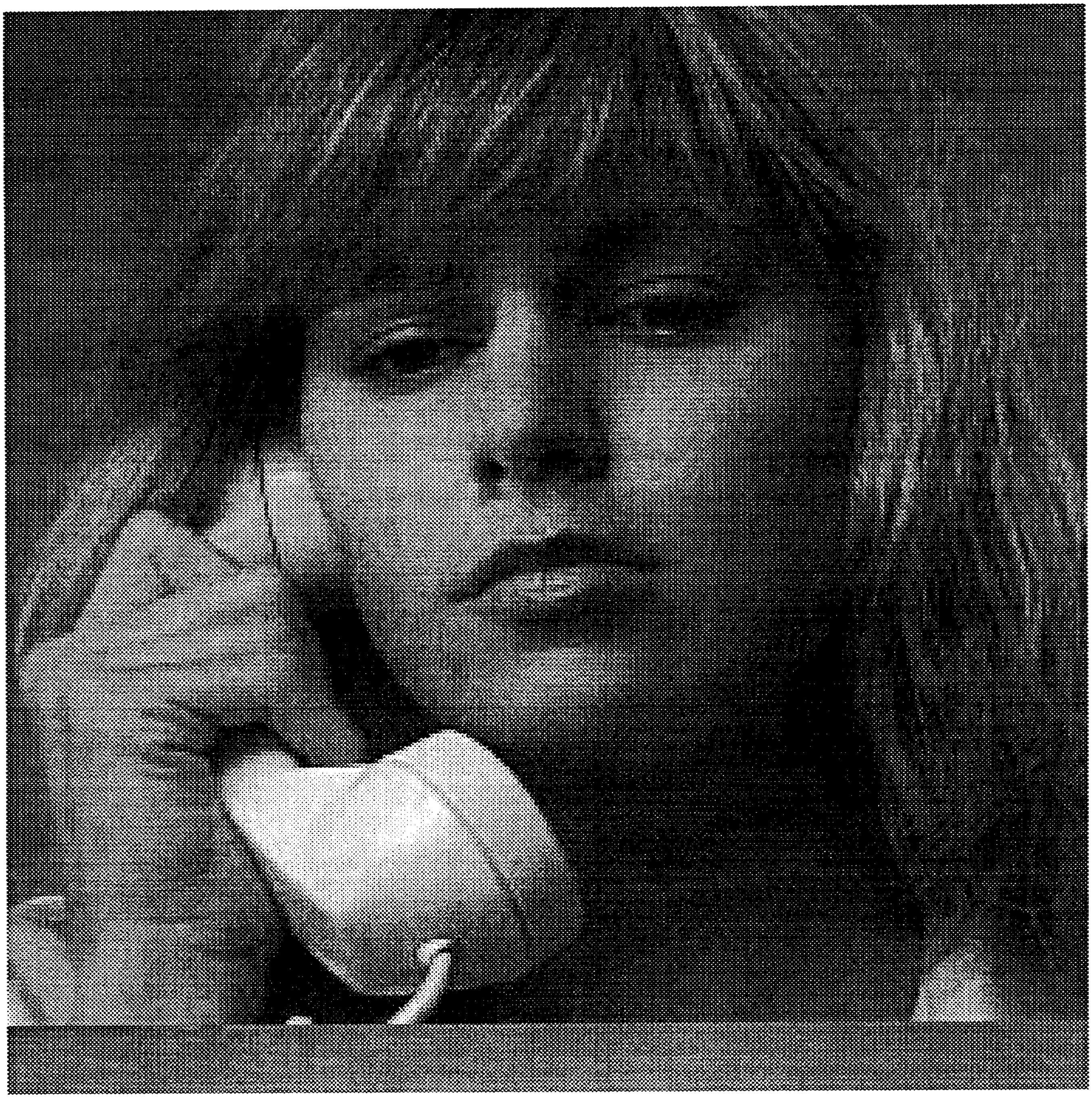


Figure 2.
Difference image



ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Figure 3.
Reconstructed image with threshold = 3 (no coding)



ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Figure 4.
Reconstructed image with threshold = 5 (no coding)

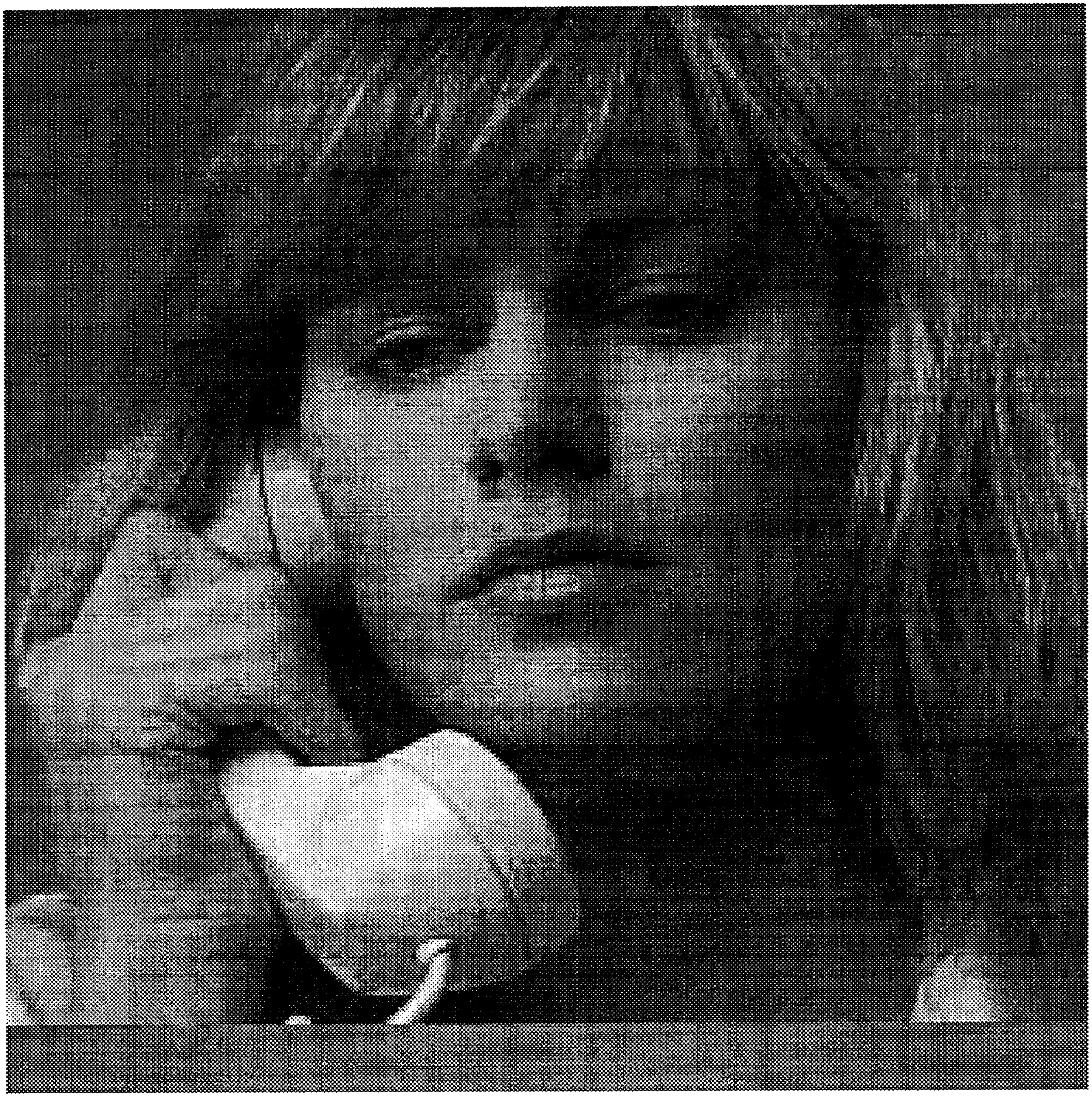


Figure 5.

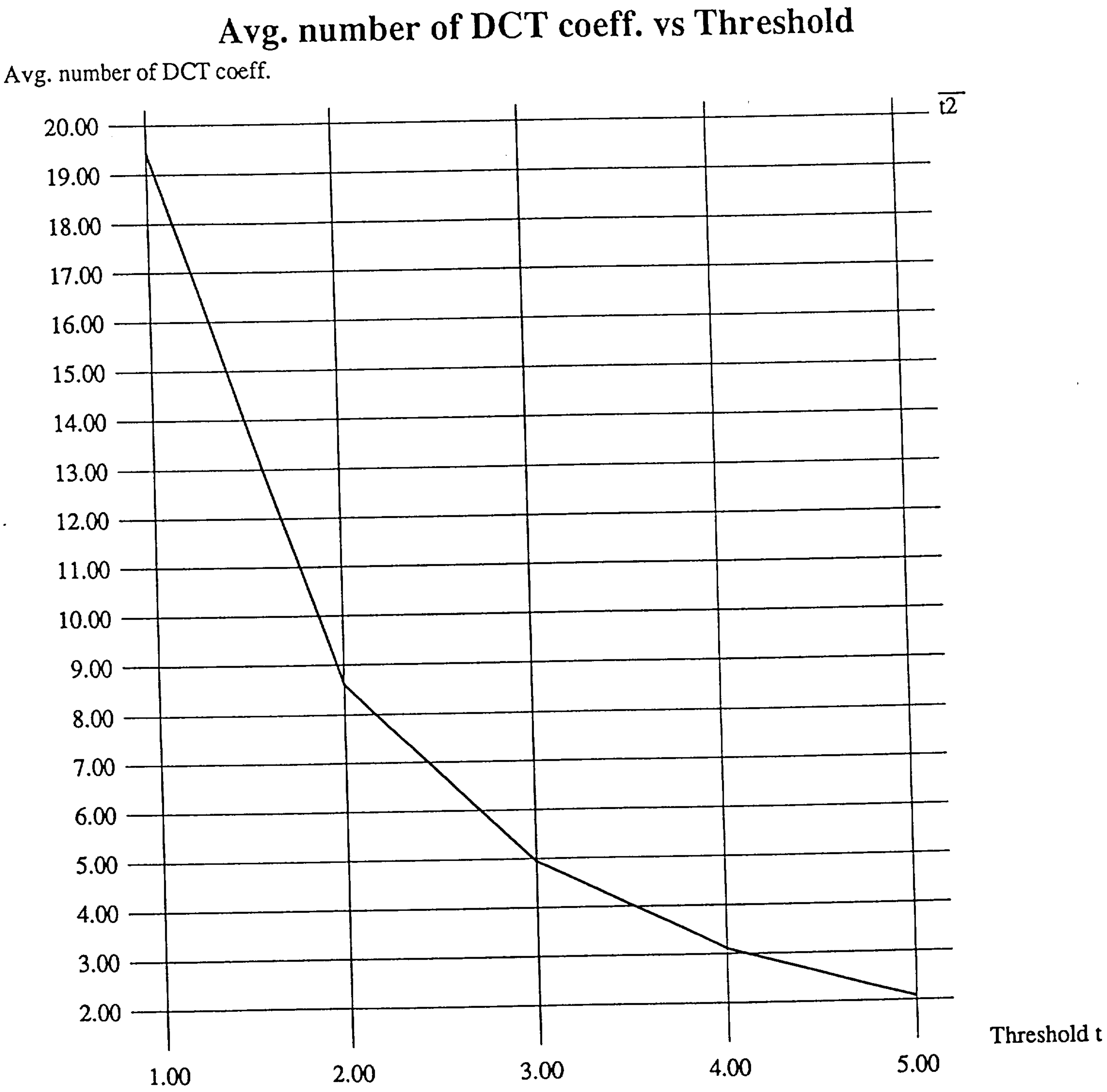

Avg. number of DCT coeff. vs Threshold
Avg. number of DCT coeff.
t2
20.00
19.00
18.00
17.00
16.00
15.00
14.00
13.00
12.00
11.00
10.00
9.00
8.00
7.00
6.00
5.00
4.00
3.00
2.00
1.00
2.00
3.00
4.00
5.00
Threshold t

Figure 6.

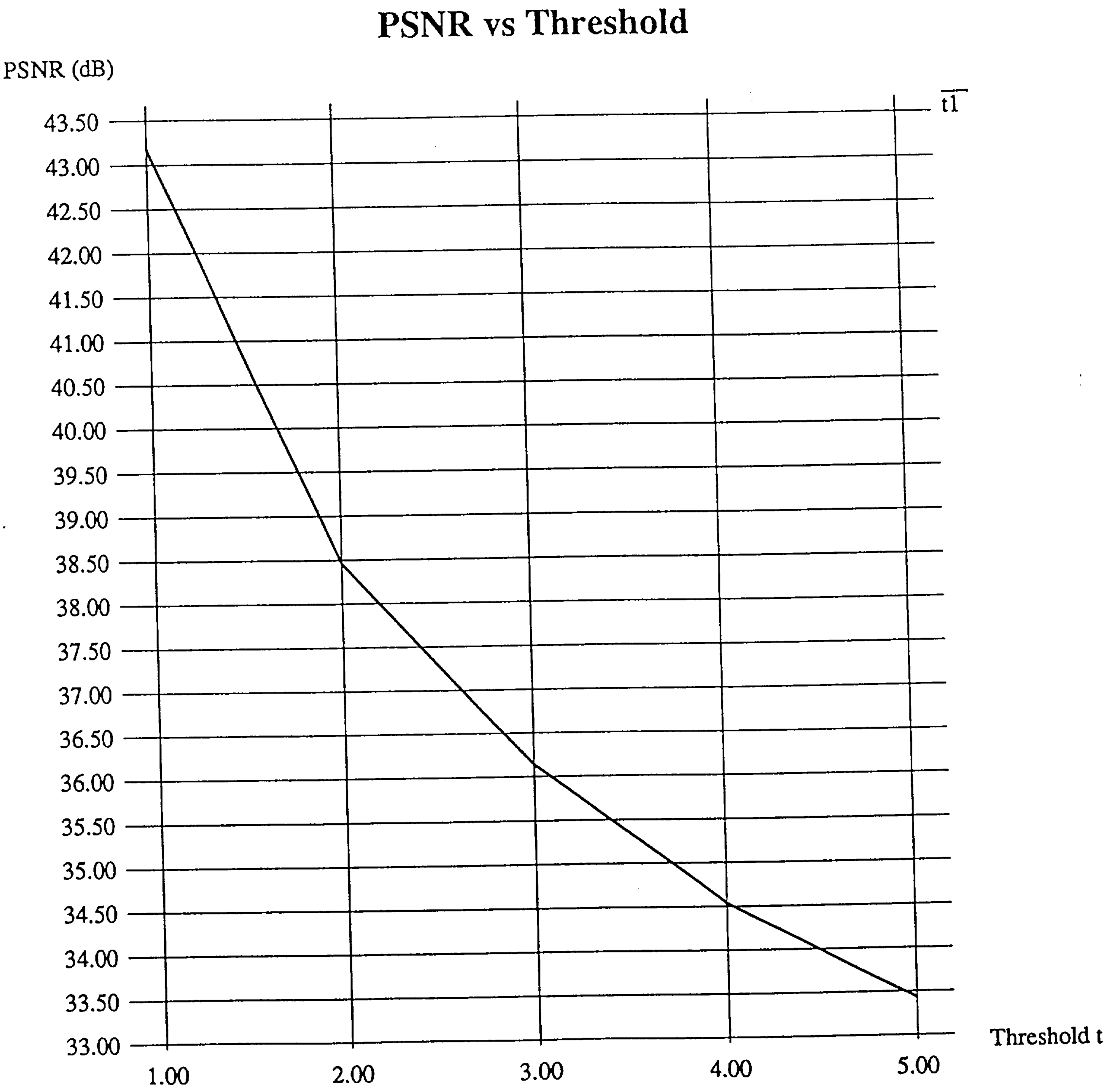
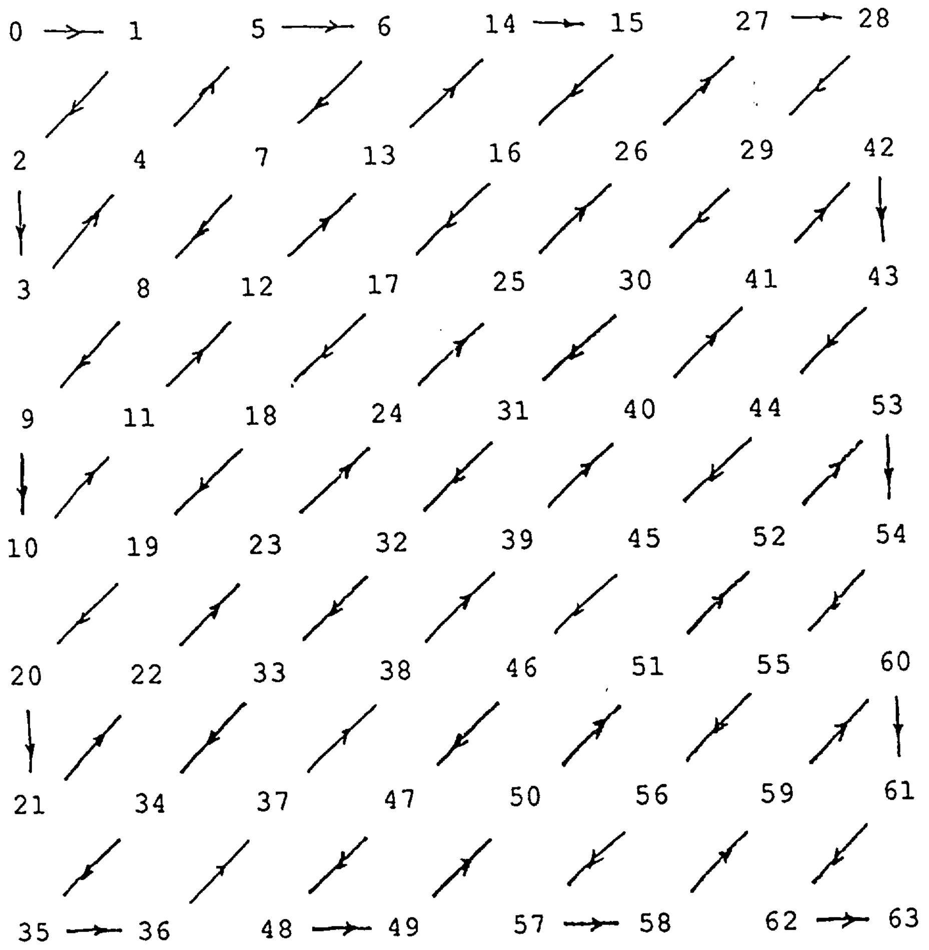
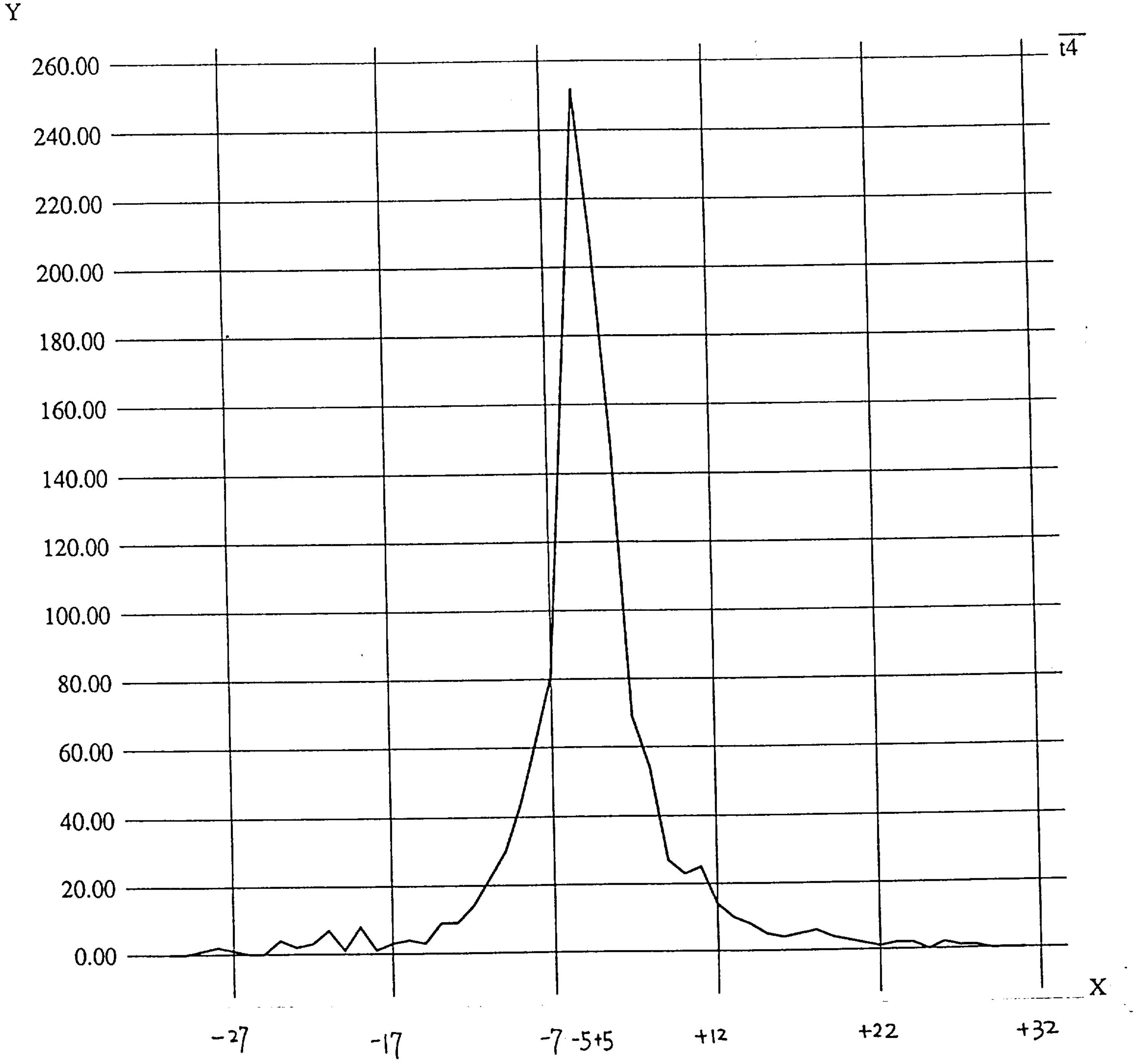

PSNR vs Threshold
PSNR (dB)
43.50
43.00
42.50
42.00
41.50
41.00
40.50
40.00
39.50
39.00
38.50
38.00
37.50
37.00
36.50
36.00
35.50
35.00
34.50
34.00
33.50
33.00
t1
Threshold t
1.00
2.00
3.00
4.00
5.00

Figure 7.
Zig-zag scanning pattern



0	1	5	6	14	15	27	28
2	4	7	13	16	26	29	42
3	8	12	17	25	30	41	43
9	11	18	24	31	40	44	53
10	19	23	32	39	45	52	54
20	22	33	38	46	51	55	60
21	34	37	47	50	56	59	61
35	36	48	49	57	58	62	63

Figure 8.
Distribution of coefficients with threshold = 5



ORIGINAL PAGE
BLACK AND WHITE PHOTOGRAPH

Figure 9.
Reconstructed image with coding rate 0.24 bpp

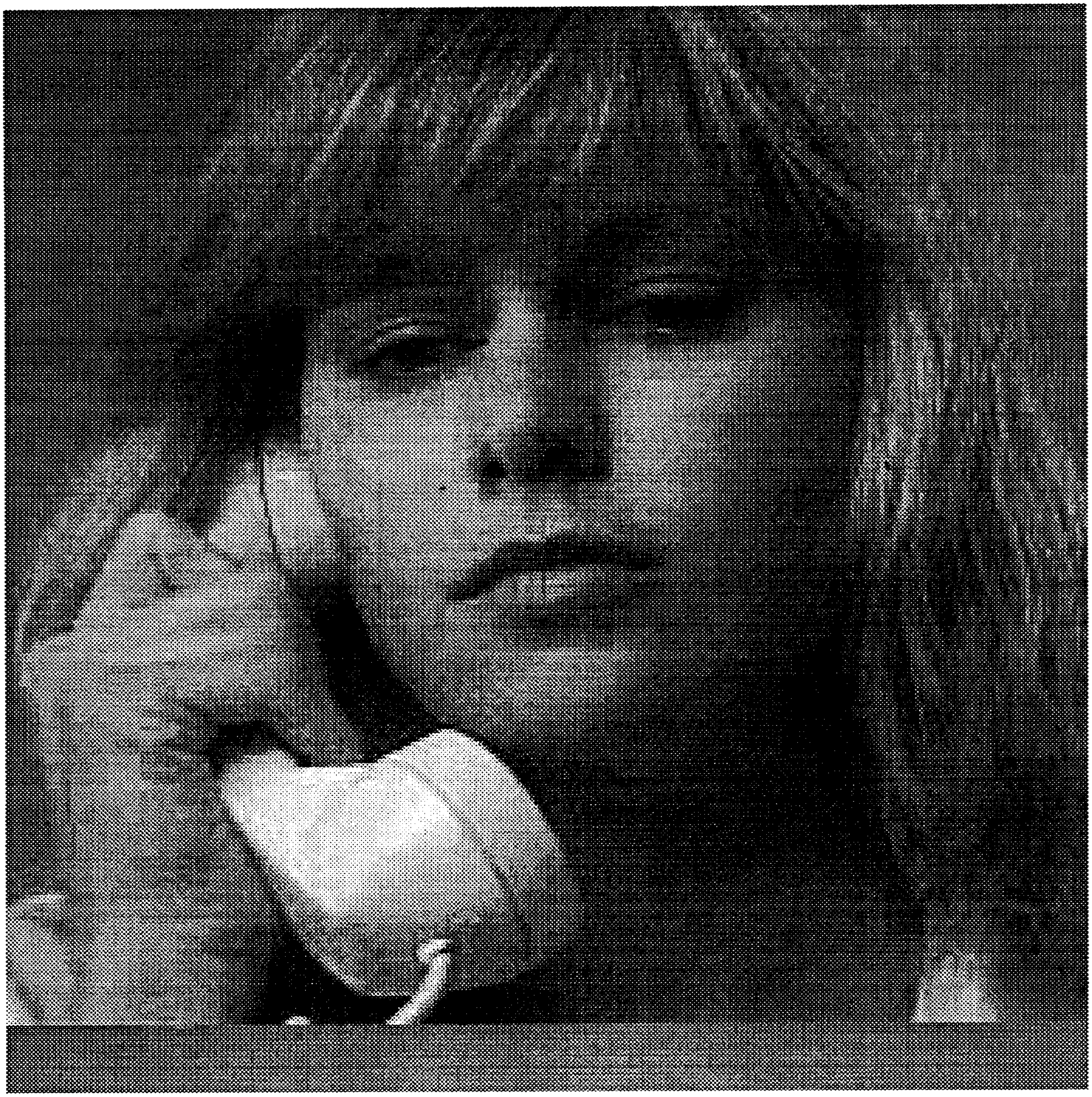


Figure 10.

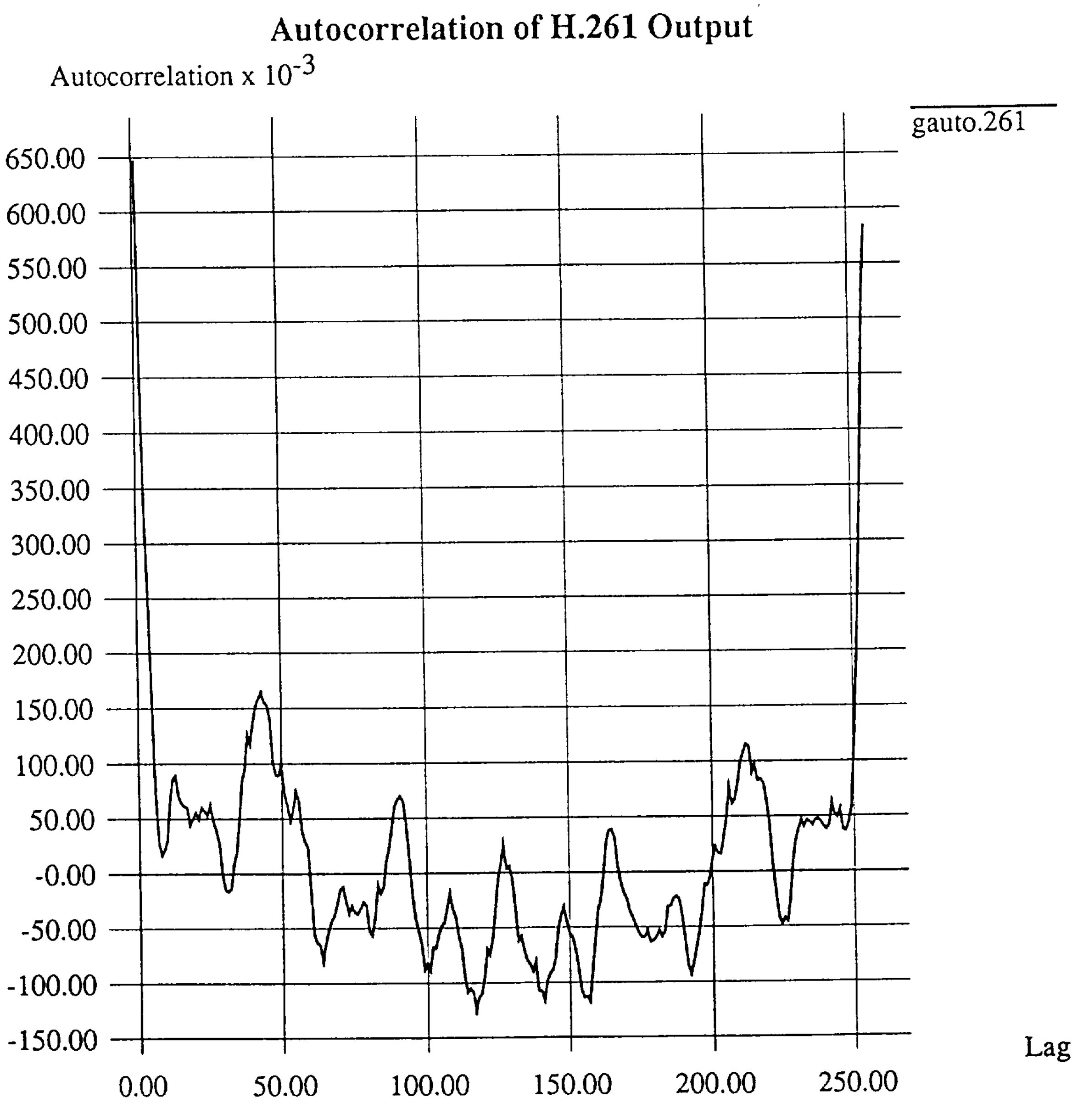
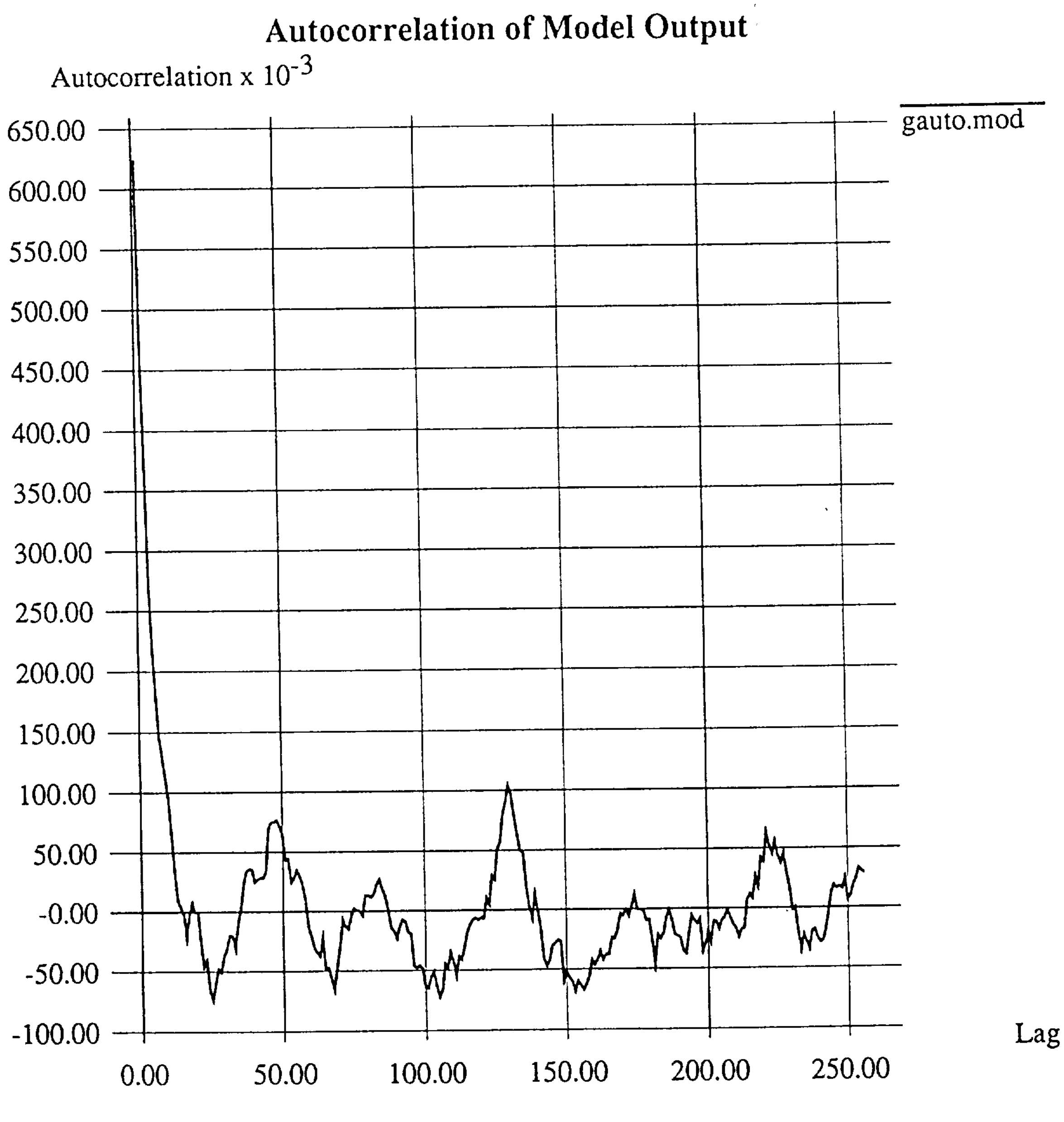

Autocorrelation of H.261 Output
Autocorrelation x 10^{-3}
gauto.261
650.00
600.00
550.00
500.00
450.00
400.00
350.00
300.00
250.00
200.00
150.00
100.00
50.00
-0.00
-50.00
-100.00
-150.00
0.00
50.00
100.00
150.00
200.00
250.00
Lag

Figure 11.


Autocorrelation of Model Output
Autocorrelation x 10^{-3}
650.00
600.00
550.00
500.00
450.00
400.00
350.00
300.00
250.00
200.00
150.00
100.00
50.00
-0.00
-50.00
-100.00
0.00
50.00
100.00
150.00
200.00
250.00
Lag
gauto.mod

Appendix

Test Images

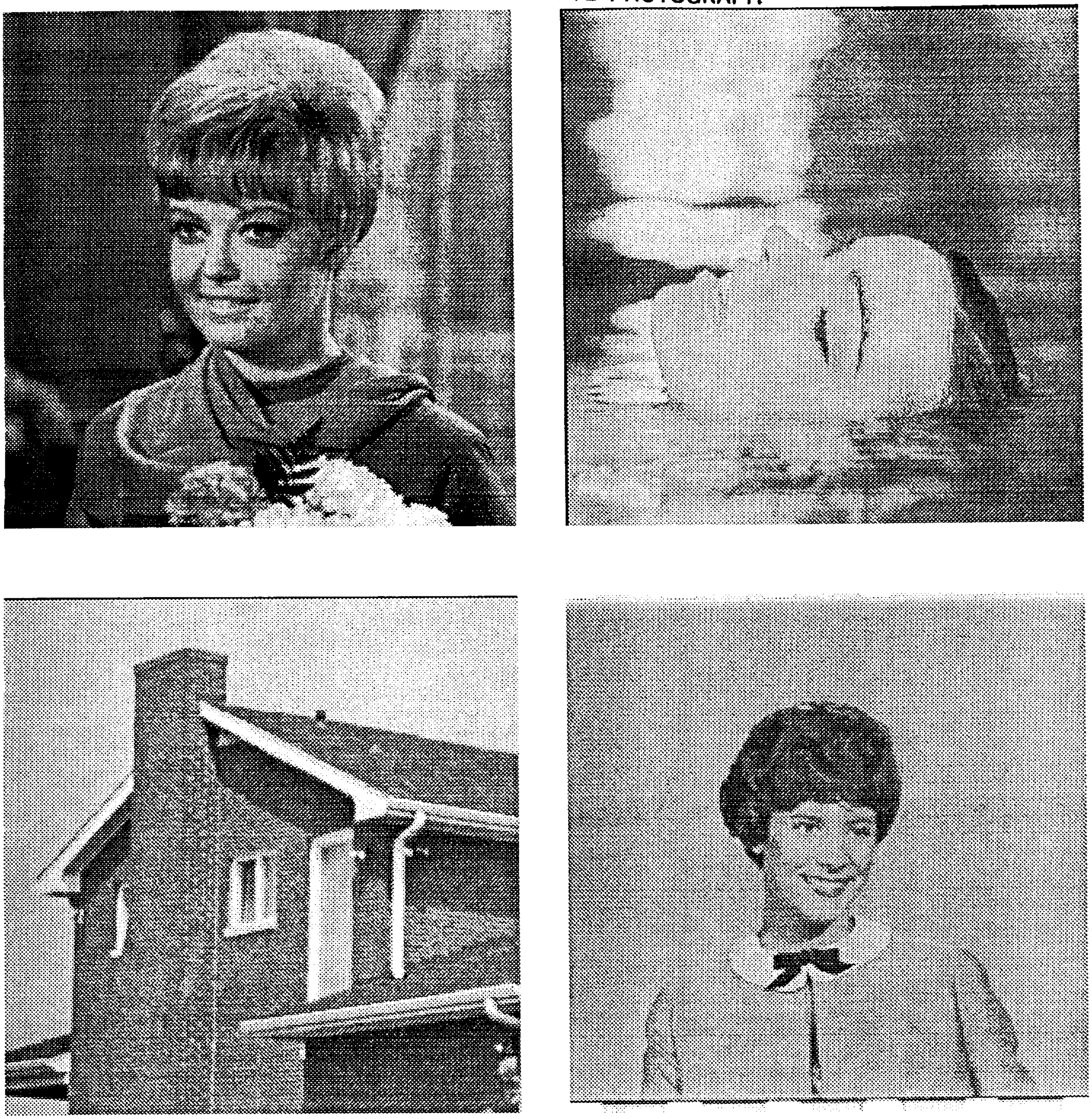


Figure 0.1: Clockwise from top left to bottom left 1) USC-Girl 2) Girl 3) Lady 4) House

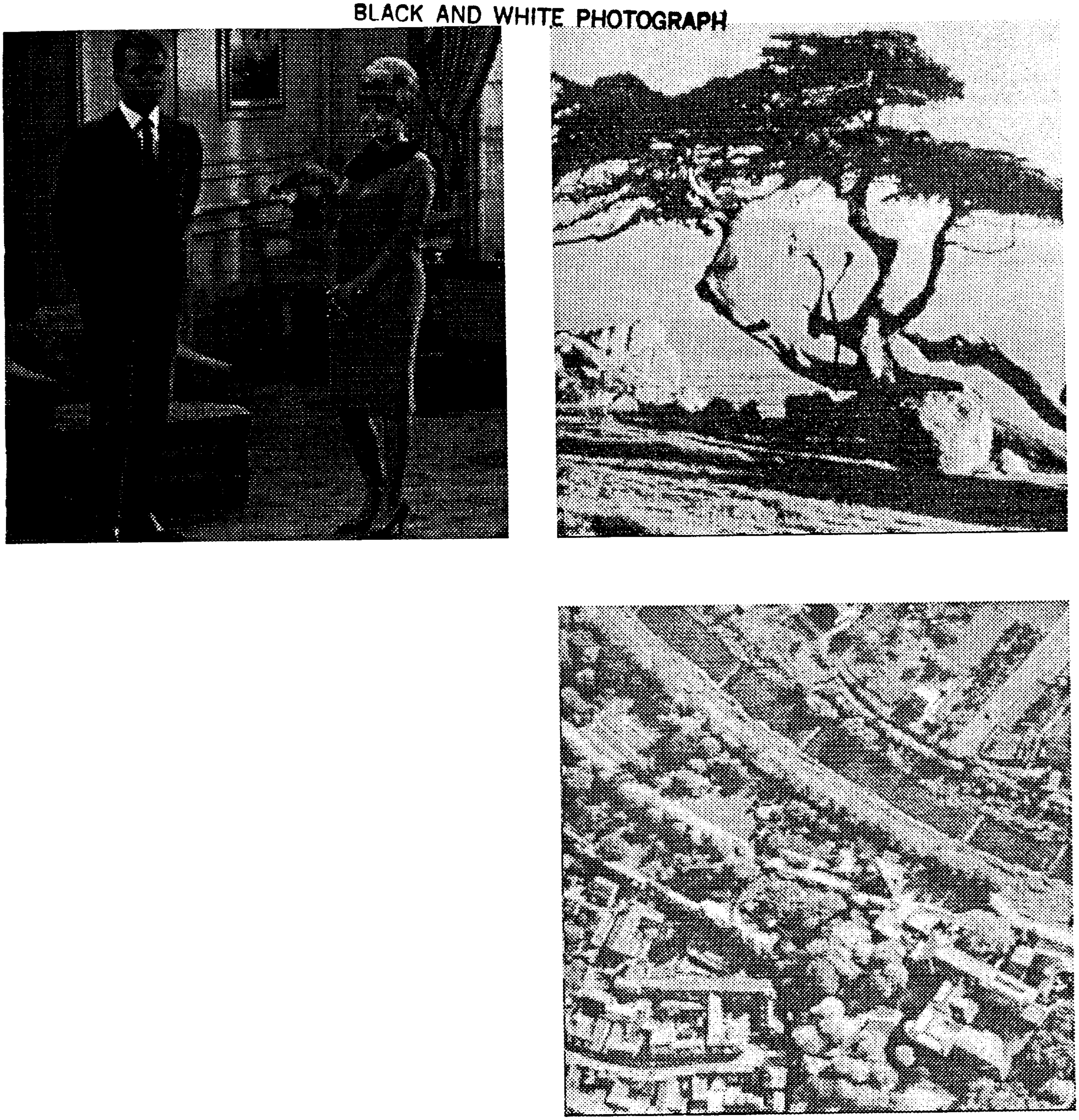


Figure 0.2: Clockwise from top left to bottom left 1) USC-Couple 2) Tree 3) Satellite

Appendix 1

Edge Preserving DPCM

An Edge Preserving Differential Image Coding Scheme

Martin C. Rost and Khalid Sayood

Abstract—**Differential encoding techniques are fast and easy to implement. However, a major problem with the use of differential encoding for images is the rapid edge degradation encountered when using such systems. This makes differential encoding techniques of limited utility especially when coding medical or scientific images, where edge preservation is of utmost importance. We present a simple, easy to implement differential image coding system with excellent edge preservation properties. The coding system can be used over variable rate channels which makes it especially attractive for use in the packet network environment.**

I. Introduction

The transmission and storage of digital images requires an enormous expenditure of resources, necessitating the use of compression techniques. These techniques include relatively low complexity predictive techniques such as adaptive differential pulse code modulation (ADPCM) and its variations, as well as relatively higher complexity techniques such as transform coding and vector quantization [1], [2]. Most compression schemes were originally developed for speech and their application to images is at times problematic. This is especially true of the low complexity predictive techniques. A good example of this is the highly popular ADPCM scheme. Originally designed for speech [3], it has been used with other sources with varying degrees of success. A major problem with its use in image coding is the rapid degradation in quality whenever an edge is encountered. Edges are perceptually very important, and therefore, their degradation can be perceptually very annoying. If the images under consideration contain medical or scientific data, the problem becomes even more important, as edges provide position information which may be crucial to the viewer. This poor edge reconstruction quality has been a major factor in preventing ADPCM from becoming as popular for image coding as it is for speech coding. While good edge reconstruction capability is an important requirement for image coding schemes, another requirement that is gaining in importance with the proliferation of packet switched networks is the ability to encode the image at different rates. In a packet switched network, the available channel capacity is not a fixed quantity, but rather fluctuates as a function of the load on the network. The compression scheme must, therefore, be capable of taking advantage of increased capacity when it becomes available while providing graceful degradation when the rate decreases to match decreased available capacity.

In this paper we describe a DPCM-based coding scheme which has the desired properties listed above. It is a low complexity scheme with excellent edge preservation in the reconstructed image. It takes full advantage of the available channel capacity providing lossless compression when sufficient capacity is available, and very graceful degradation when a reduction in rate is required.

II. Notation and Problem Formulation

The DPCM system consists of two main blocks, the quantizer and the predictor (see Fig. 1). The predictor uses the correlation between samples of the waveform $s(k)$ to predict the next sample value. This predicted value is removed from the waveform at the transmitter and reintroduced at the receiver. The prediction error is quantized to one of a finite number of values which is coded and transmitted to the receiver and is denoted by $e_q(k)$. The difference between the prediction error and the quantized prediction error is called the quantization error or the quantization noise. If the channel is error free, the reconstruction error at the receiver is simply the quantization error. To see this, note (Fig. 1) that the prediction error $e(k)$ is given by

$$e(k) = s(k) - p(k) \tag{1}$$

Manuscript received January 18, 1990; revised April 28, 1991. This work was supported by the NASA Goddard Space Flight Center under Grant NAG-5-916.

M. C. Rost is with Sandia National Laboratories, Albuquerque, NM 87185.

K. Sayood is with the Department of Electrical Engineering and the Center for Communication and Information Science, University of Nebraska, Lincoln, NE 68588-0511.

IEEE Log Number 9106076.

with time. The actual change can be accommodated by changing the stepsize and reducing the lossless encoder codebook size by the same amount. Several of the systems proposed above were simulated. The results of these simulations are presented in the next section.

IV. Results

Before we provide the results using images, let us examine the performance of the scheme when applied to a one-dimensional signal containing a simulated edge. This signal was first encoded using a five-level quantizer. The results are shown in Fig. 3(a). As can be seen, it takes a little while for the DPCM system to catch up. In an image this would cause a smearing of the edge. When the proposed system with the same parameters is used there is no such effect, as is clear from Fig. 3(b). The quantizer in this case went into the recursive mode twice, once at the leading and once at the trailing edge. To get an equivalent effect, a standard DPCM system would have to have a forty-level quantizer. To show that this performance is maintained when the system is used with two-dimensional images, two systems of the type described in the previous section have been simulated. Both systems use the following two-dimensional fixed predictor [7]: $p(k) = 2/3\hat{s}(k - 1) + 2/3\hat{s}(k - 256) - 1/3\hat{s}(k - 257)$. One of the systems contains the lossless encoder followed by a runlength encoder while the other contains only the lossless encoder without the runlength encoder. The test images used were the USC GIRL image, and the USC COUPLE image. Both are 256 by 256 monochrome 8-b images and have been used often as test images. The objective performance measures were the peak signal-to-noise ratio (PSNR) and the mean absolute error (MAE) which are defined as follows:

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\langle (s(k)) - \hat{s}(k)^2 \rangle}$$

$$\mathrm{MAE} = \langle |s(k) - \hat{s}(k)| \rangle$$

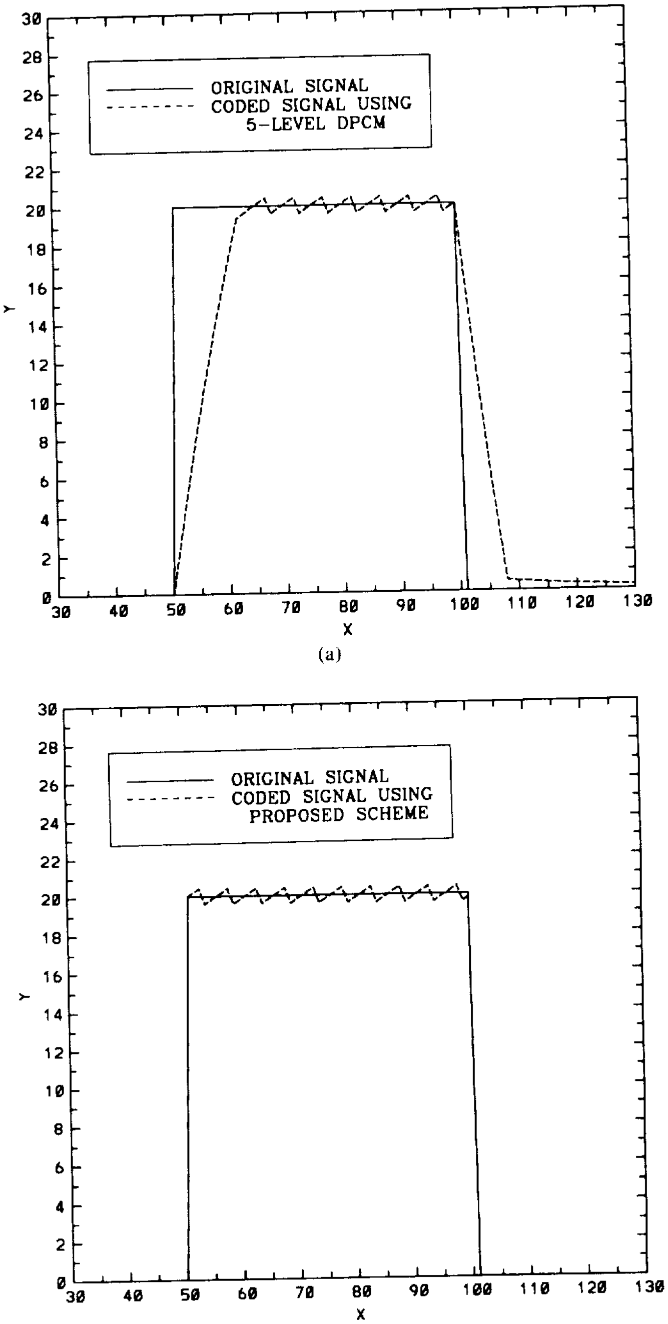
where $\langle \cdot \rangle$ denotes the average value.

Several initial test runs were performed using a different number of levels, different values of x_L, and different values of Δ to get a feel for the optimum values of the various parameters (given x_L and Δ, x_H is automatically determined). We found that an appropriate way of selecting the value of x_L was using the relationship

$$x_L = -\left\lfloor \frac{N-1}{2} \right\rfloor \Delta$$

where $\lfloor x \rfloor$ is the largest integer less than or equal to x, and N is the size of the alphabet of the lossless coder. This provides a symmetric codebook when the alphabet size is odd, and a codebook skewed to the positive side when the alphabet size is even. The zero value is always in the codebook.

As the alphabet size is usually not a power of two, the binary code for the output alphabet will be a variable length code. The use of variable length codes always bring up issues of robustness with respect to changing input statistics. With this in mind, the rate was calculated in two different ways. The first was to find the output entropy, and scale it up by the ratio of symbols transmitted to the number of pixels encoded. We call this rate the entropy rate, which is the minimum rate obtainable if we assume the output of the lossless encoder to be memoryless. While this assumption is not necessarily true, the entropy rate gives us an idea about the best we can do with a particular system. We also calculated the rate using a predetermined variable length code. This code was designed with no prior knowledge of the probabilities of the different letters. The only assumption was that the letters representing the inner levels of the quantizer were always more likely than the letters representing the outer levels of the quantizer. The code tree used is shown in Fig. 4. Obviously, this will become highly inefficient in the case of small alphabet size and small Δ, as in this case, the outer levels x_L and x_H will occur quite frequently. This rate can be viewed as an upper bound on the achievable rate.

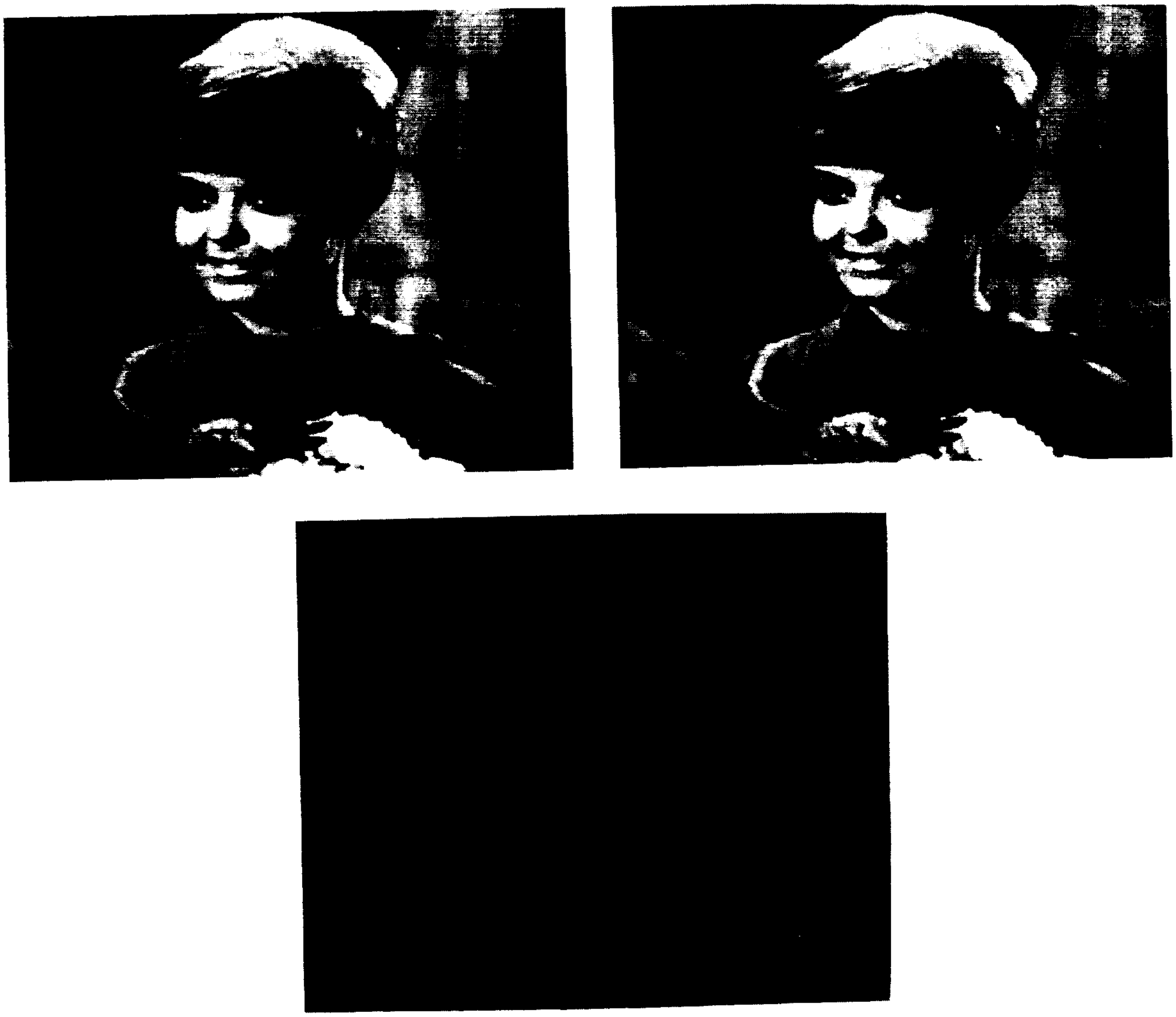


(a)

(b)

Fig. 3. Coding of simulated one-dimensional edge with (a) DPCM, (b) proposed system.

The results for the system without the runlength encoder are shown in Tables I and II. Table I contains the results for the COUPLE image, while Table II contains the results for the GIRL image. In the table R_L denotes the entropy rate while R_U is the rate obtained using the Huffman code of Fig. 4. Recall that for image compression schemes, systems with PSNR values of greater than 35 dB are perceptually almost identical. As can be seen from the PSNR values in the tables there is very little degradation with rate, and in fact, if we use the 35-dB criterion, there is almost no degradation in image quality until the rate drops below 2 b/pixel. This can be verified by the reconstructed images shown in Fig. 5. Each picture



(a)

Fig. 5(a). GIRL image coded at entropy rate of 1.5 bpp. (b) GIRL image coded at entropy rate of 1.3 bpp.

in Fig. 5 consists of the original image, the reconstructed image and the error image magnified 10 fold. In each of the pictures, it is extremely difficult to tell the source or original image from the reconstructed or output image. This subjective observation is supported by the error images in each case which are uniform in texture throughout without the edge artifacts which can be usually seen in the error images for most compression schemes.

We can see from the results that if the value of Δ, and hence, x_l is fixed, the size of the codebook has no effect on the performance measures. This is because the only effect of reducing the codebook size under these conditions is to increase the number of symbols transmitted. While this has the effect of increasing the rate, because of the way the system is constructed it does not influence the resulting distortion. The drop in rate for the same distortion as the alphabet size increases can be clearly seen from the results in Tables I and II.

Table III and Table IV show the decrease in rate when a simple runlength coder is used. The runlength coder encodes long strings of x_L and x_H using the special sequences mentioned previously. As can be seen from the results the improvement provided by the current runlength encoding scheme is significant only for small alphabets and small values of Δ. This is because it is under these conditions that most of the long strings of x_L and x_H are generated. However, we are not as yet using many of the special sequences in the larger alphabet codebooks, so there is certainly room for improvement.

Finally to show the effect of changing rate on the perceptual quality, the USC GIRL image was encoded using three different rates. The top quarter of the image was encoded using a codebook size of eight and a Δ of two resulting in a rate of 4.37 b/pixel. The second quarter of the image was encoded using a codebook of size five and a Δ of 4 resulting in a rate of 2.86 b/pixel. The bottom half of the image was encoded using a codebook size of three and Δ of eight resulting in a rate of 2.36 b/pixel. The original and reconstructed images are shown in Fig. 6. The fact that the image is coded with three different rates can only be noticed if the viewer is already aware of this fact and then only after very close scrutiny. The fact that the image was encoded using three different rates is clear in the magnified error image shown in Fig. 7. This property of the coding scheme would be extremely useful if changes in the transmission bandwidth forced the coder to operate at different rates.

To see how this algorithm performs on a relative scale, we compare it to the differential scheme proposed by Maragos, Shafer, and Mersereau [8]. The system proposed by Maragos *et al.* uses a forward adaptive two-dimensional predictor and a backward adaptive

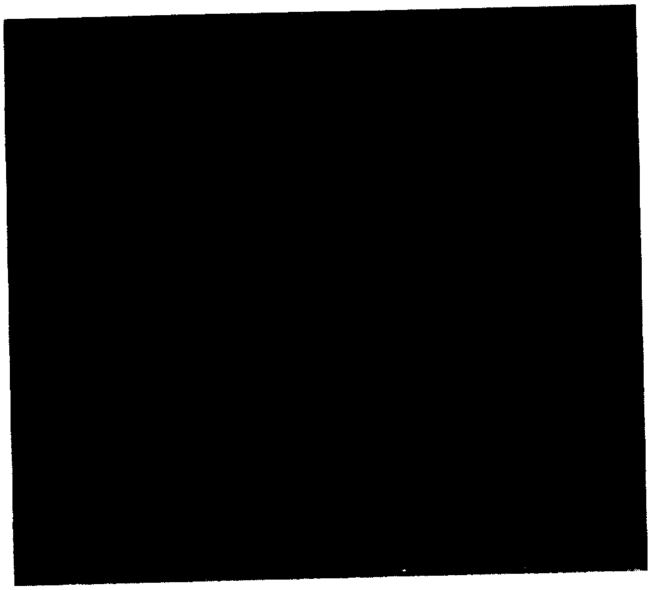


Fig. 7. Error image for GIRL image coded at three different rates.

TABLE V
COMPARISON OF PROPOSED SYSTEM WITH THAT OF [8]

Results from [8] (Frame Size = 32, 3 Level AQB)		Results from Proposed System (Alphabet Size 5)	
Rate	PSNR	Rate	PSNR
0.74	30.3	0.74	31.13
0.83	31.6	0.84	32.1
0.93	32.6	0.94	33.1
1.03	33.4	1.03	33.9

quantizer. The coefficients are obtained over a 32 by 32 or a 16 by 16 block and transmitted as side information. The proposed system (we feel) is considerably simpler, because of the lack of any need for adaptation and side information; however, the results compare favorably with the system of [8]. Comparative results are shown in Table V. The results were obtained by varying the stepsize Δ until the rate obtained was similar to the rate in [8], and then comparing the PSNR. As in [8], to obtain rates below 1 b/pixel, several coder outputs were concatenated into blocks which were then Huffman encoded. For the results shown in Table V, we used a block size of three. Given a five-level recursive quantizer, this corresponds to an alphabet size of 125, which would be somewhat excessive for a simple implementation. (In [8] block sizes of four to eight are used with two- and three-level quantizers.)

The above comparison is not meant to indicate that the two systems being compared are exclusive. A case can be made for combining the good features of both systems. For example, the prediction scheme described in [8] could be combined with the quantization scheme described here. However, it was felt in this particular case that the advantages to be gained by the addition of a forward adaptive predictor were offset by the increase in complexity and synchronization requirements.

V. CONCLUSION

We have demonstrated a simple image coding scheme which is very easy to implement in real time and has excellent edge preservation properties over a wide range of rates.

This system would be especially useful in transmitting images over channels were the available bandwidth may be vary. The edge preserving quality is especially useful in the encoding of scientific and medical images.

REFERENCES

[1] N. S. Jayant and P. Noll, *Digital Coding of Waveforms*. Englewood Cliffs, NJ: Prentice-Hall, 1984.

[2] A. K. Jain, "Image data compression: A review," *Proc. IEEE*, vol. 69, pp. 349-389, Mar. 1981.

[3] C. C. Cutler, "Differential quantization for communication signals," U.S. Patent 2 605 361, July 29, 1952.

[4] J. D. Gibson, "Backward adaptive prediction as spectral analysis within a closed loop," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. ASSP-33, pp. 1166-1174, Oct. 1985.

[5] K. Sayood and S. M. Schekall, "Use of ARMA predictors in the differential encoding of images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-36, pp. 1791-1795, Nov. 1988.

[6] S. M. Schekall and K. Sayood, "An edge preserving DPCM scheme for image coding," *Proc. 31st Midwest Symp. Circuits Systems*, St. Louis, pp. 904-907, Aug. 1988.

[7] C. W. Harrison, "Experiments with linear prediction in television," *Bell Syst. Tech. J.*, vol. 31, pp. 764-783, July 1952.

[8] P. A. Maragos, R. W. Schafer, and R. M. Mersereau, "Two-dimensional linear prediction and its application to adaptive predictive coding of images," *IEEE Trans. Acoust. Speech, Signal Processing*, vol. ASSP-32, pp. 1213-1229, Dec. 1984.